AI-ESTATE–THE NEXT GENERATION

John Sheppard
ARINC
2551 Riva Road
Annapolis, MD 21401
410-266-2099
jsheppar@arinc.com

Mark Kaufman
NWAS
PO Box 5000
Corona, CA 91718
210-522-2190
kaufman.mark@corona.navy.mil

Abstract - In this paper, we present an overview of the current and future directions of the AI-ESTATE standards. We address the concerns of document organization, information modeling, service versus API specification, and other issues raised by the AI-ESTATE community. We also discuss the vision of the AI-ESTATE subcommittee in its work to integrate the AI-ESTATE information models and projects such as testability/diagnosability assessment and test/maintenance feedback.

INTRODUCTION

In 1998, the third of a series of three standards was published by the IEEE addressing issues in system-level diagnostics. The Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE) family of standards is now complete. Or is it? IEEE Std 1232-1995 defines the architecture of an AI-ESTATE-conformant system and has been published as a "full-use" standard; however, this standard was published before the vision of AI-ESTATE and fully gelled. IEEE Std 1232.1-1997 defines a knowledge and data exchange standard and was published as a "trial-use" standard. Trial-use indicates that it is preliminary in nature, and the standards committee is seeking comments from organizations attempting to implement or use the standard. In 1998, IEEE Std 1232.2-1998 was approved. Its publication, also as a "trial-use" standard is imminent. This standard formally defines a set of standard software services to be provided by a diagnostic reasoner in an open-architecture test environment. Since it is also a trial-use standard, comment and feedback is necessary here as well.

Following the completion of the P1232.2 ballot, the AI-ESTATE subcommittee began examining the structure of the standards to determine if a better organization existed to make the standard easier to understand and to implement. A unified version of the three standards is large and complex. The standards were developed using information modeling, resulting in the definition of four information models addressing static and dynamic aspects of the diagnostic domain. In addition, several issues have been identified by members of the AI-ESTATE subcommittee and organizations attempting to implement the standards.

In this paper, we present an overview of the current and future directions of the AI-ESTATE standards. We address the concerns of document organization, information modeling, service versus Applications Program Interface (API) specification, and other issues raised by the AI-ESTATE community. We also discuss the vision of the AI-ESTATE subcommittee in its work to integrate the AI-ESTATE information models and projects such as testability/diagnosability assessment and test/ maintenance feedback.

THE AI-ESTATE ARCHITECTURE

According to IEEE Std 1232-1995, the AI-ESTATE architecture is "a conceptual model" in which "AI-ESTATE applications may use any combination of components and intercomponent communication" [1]. On the other hand, according to IEEE Std 1232.2-1998, AI-ESTATE includes explicit definitions of services to be provided by a diagnostic reasoner, where the services "can be thought of as responses to client requests from the other components of the system architecture" [2]. More specifically, "each of the elements that interface with the reasoner will interact through [an] application executive and will provide its own set of encapsulated services to its respective clients" [2].

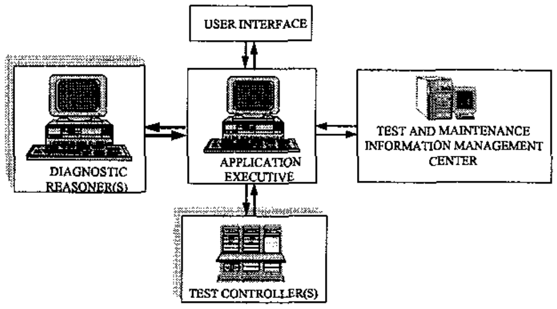

Figure 1. AI-ESTATE Architecture

Although not necessarily obvious from the standards themselves, these two "views" of the AI-ESTATE architecture present an interesting dichotomy. Specifically, the architecture standard provides a concept of AI-ESTATE that permits any communication mechanism to be used between components of a test environment in support of the diagnostics provided by that environment. The service specification, on the other hand, seems to cast the communication mechanism in the form of a client-server architecture.

We note that the intent of AI-ESTATE is to provide a formal, standard framework for the exchange of diagnostic information (both static and dynamic) in a test environment. This exchange occurs at two levels. At the first level, data and knowledge is exchanged through a neutral exchange format, as specified by IEEE Std 1232.1-1997 [3]. At the second level, information is exchanged as needed between software applications within the test environment. This information includes entities as read in from a model or information on the current state of the diagnostic process.

To facilitate encapsulation of the information and the underlying mechanisms providing that encapsulation, AI-ESTATE assumes the presence of an "application executive." We emphasize that this application executive need not be a physically separate software process but can be identified as a "view" of the software process when it involves the communication activity. This view of the architecture is shown in Figure 1. In the following sections, we will provide a more detailed discussion of the exchange and service elements of the architecture, followed by a discussion of the issues involved to ensure AI-ESTATE is a viable standard for next generation test systems.

Data and Knowledge Exchange

ISO 10303–11 (EXPRESS) and ISO 10303–12 (EXPRESS-I) are used to define information models and exchange formats for diagnostic knowledge [4], [5]. These international standards being maintained by the STEP (Standard for the Exchange of Product model data) community. The current approach to static information exchange within AI-ESTATE is to derive the exchange format from the formal information models as specified in the ISO standards.

The purpose of information modeling is to provide a formal specification of the *semantics* of information that is being used in an "information system." Specifically, information models identify the key entities of information to be used, their relationships to one another, and the "behavior" of these entities in terms of constraints on valid values [6]. The intent is to ensure that definitions of these entities are unambiguous.

For example, central to the test and diagnosis problem is the definition of a "test." If we ask a digital test engineer what a test is, it is possible that the answer will be something like "a set of vectors used to determine whether or not a digital circuit is working properly." On the other hand, if we ask a diagnostic modeler what a test is, the answer is likely to be "any combination of stimulus, response, and a basis for comparison that can be used to detect a fault."

On the surface, these two definitions appear very similar; however, there is a fundamental difference. For the digital test engineer, there is an implicit assumption that a "test" corresponds to the entire suite of vectors. For the diagnostic modeler, individual vectors are tests as well.

As a similar example, the test engineer and diagnostic modeler are likely to have different definitions for "diagnosis." The act of doing diagnosis, for most test engineers, corresponds to running tests after dropping off of the "go-path." For the diagnostic modeler, since "no fault" is a diagnosis, the entire test process (including the go-path) is part of doing diagnosis.

It may appear that we are "splitting hairs," but formal definition of terms and information entities is an exercise in splitting hairs. Further, such hair-splitting is essential to ensure that communication is unambiguous—especially when we are concerned with communication between software processes. No assumption can go unstated; otherwise, the risk exists

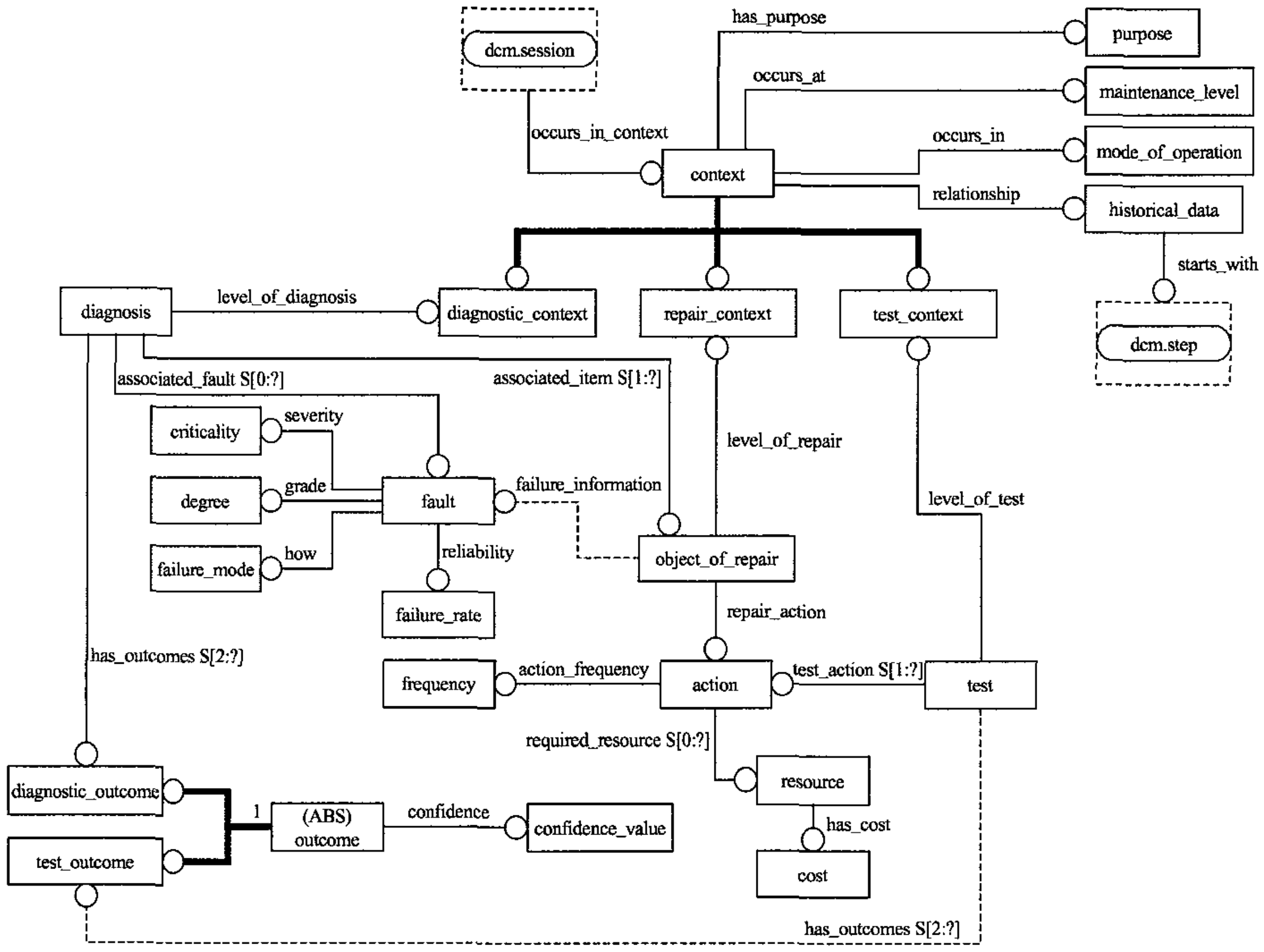

Figure 2. Revised Common Element Model

that something will be misunderstood. Information models formally state all of the assumptions.

A New Information Model

When IEEE 1232.1 was published, it was published as a "trial-use" standard to provide a period for people to study it, attempt to implement it, and provide feedback to the AI-ESTATE committee on the ability of the standard to satisfy the stated requirements. Since publication, comments have been received to indicate that ambiguity still exists in the information models.

Because of the concern that the information models are still ambiguous, the models are undergoing close examination and modification. It is interesting to note that much of the ambiguity has been identified in connection with a related standard being developed by the AI-ESTATE committee—P1522 Standard for Testability and Diagnosability Metrics and Characteristics. AI-ESTATE's approach to developing this new standard involved defining the metrics based on the information models within the P1232 standard. As we were identifying metrics to be standardized, we discovered that the current models were incapable of supporting their definition.

A conceptual view of the revised common element model is shown in Figure 2. To support this revised model, additional reorganization of the entities in the model is underway (as shown in Figure 3).

Of note in the revised model is the addition of a context entity and the differentiation between fault and function. Many diagnostic tools are highly context dependent (e.g., different procedures are suggested based on the environmental conditions of the test or the skill levels of the maintenance technicians). In

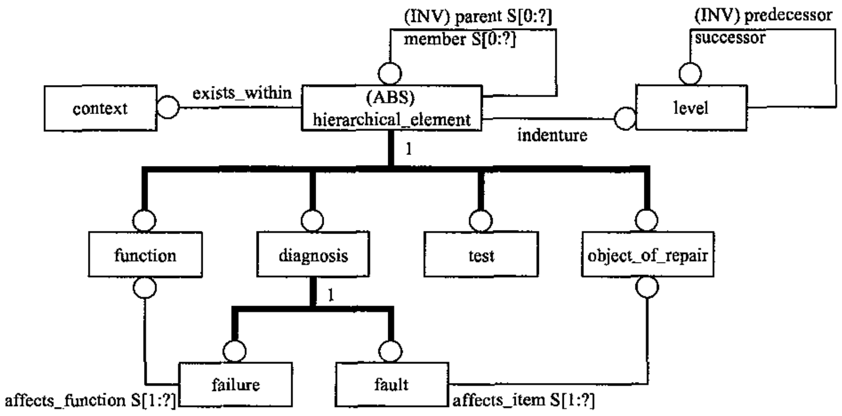

Figure 3. Function/Fault Information Model

addition, several tools focus on modeling function rather than physical faults to support modeling at the system level. Since the distinctions among context and type of analysis were not previously made explicit, new entities were defined to eliminate ambiguity that may arise from different approaches and contexts for modeling.

Diagnostic Services

The approach taken to defining services in AI-ESTATE has been based on the traversal (i.e., the following of the relationships defined between model entities to access specific pieces of information in the models) of the information models. The "simplest" services involve traversing the models defined in IEEE 1232.1 (i.e., the exchange models); however, these models provide little functionality in terms of actual diagnosis.

In IEEE 1232.2, a novel use of information modeling was applied in that a dynamic information model was specified to support dynamic services. This model, called the "dynamic context model," relied on dynamically creating entities that populate the model during a diagnostic session. In fact, as suggested by "dcm.session" and "dcm.step" in the model shown in Figure 2, a diagnostic session is modeled as a sequence of steps instantiating from the set of possible values specified in the static model. Details of how the service specification is expected to be implemented can be found in [7], [8].

One of the concerns raised by a member of the AI-ESTATE committee was whether the standard specifies a set of services or simply an API. The claim was that the service specification must include a behavior specification as well and that this can only be accomplished by defining a set of baseline behaviors, perhaps through some sort of test bed.

The committee observed that people have different opinions over the difference between a service specification and an API specification. Many, in fact, took issue with the claim that they were different. Further, it was determined that including test cases to specify standard behavior was not desirable in this context due to the wide variety of diagnostic approaches uses common diagnostic knowledge. Rather, it was believed that it was more important for the information itself to be standardized and the specific behavior to be left to the implementation.

A VISION FOR TEST AND DIAGNOSIS STANDARDS

The vision of AI-ESTATE is to provide an integrated, formal view of diagnostic information as it exists in diagnostic knowledge bases and as it is used (or generated) in diagnostic systems. We assert that the whole purpose of testing is to perform diagnosis [9]. In justifying this assumption, we rely on a very general definition of diagnosis, derived from its Greek components (δια γιγνωσκω) meaning, "to discern apart." Given such a broad definition, all testing is done to provide information about the object being tested and to differentiate some state of that object from a set of possible states.

In support of this vision, the AI-ESTATE committee has been working on combining the existing standards into a single, cohesive standard. This "unified" standard provides formal specifications of all of the information models (both for file exchange and for diagnostic processing), from which the service specifications are then derived and specified. The architectural framework is retained at the conceptual level to emphasize that a wide variety of implementation models are possible that still support standard exchange of information as long as the definition of that information is clear and unambiguous. Thus, in a sense, the models define the architecture, and the implementation is left entirely to the implementer.

With this vision in mind, we believe AI-ESTATE plays a central role in any test environment (thus the "All Test Environments" part of the name). To date, the focus of the standards has been the development of specifications supporting diagnosis in the traditional sense of the word (i.e., fault isolation). However, the broader context within which AI-ESTATE is envisioned to participate involves tying diagnostic information to explicit product behavior descriptions, assessments of the ability of testing to satisfy its requirements, and maturation of the diagnostic process through test and maintenance information feedback.

Ties to Testability

In 1997, the AI-ESTATE committee began to work on a new standard focusing on replacing the cancelled MIL-STD 2165 [10]. The military standard focused on specifying the essential elements of a testability program and explained the elements needed to define a testability program plan. In addition, MIL-STD 2165 included the "definition" of several testability metrics, including a testability checklist to be used to determine overall system testability. With the cancellation of military standards and specifications by the Perry Memo in 1994 [11], and with the lack of specificity and clarity in MIL-STD 2165, it became evident that a replacement was necessary.

The approach being taken to develop this standard involves defining testability and diagnosability metrics based on standard information models. Specifically, it was found that the AI-ESTATE models provided an excellent foundation for defining these metrics. As an example, one metric defined using the model is Fractions of Faults Detected (FFD).

The FFD metric assumes the existence of a diagnostic model that ties tests (especially test outcomes) to potential faults in the system analyzed. Within AI-ESTATE, tests, diagnoses, and faults are modeled explicitly in the common element model. In addition, AI-ESTATE includes specifications for two diagnostic models—the fault tree model and the Enhanced Diagnostic Inference Model (EDIM). Due to its generality, the EDIM was used to define FFD.

The assumptions used to define FFD are as follows:

- We are interested in the various metrics at a particular level;
- A hierarchical element exists at a particular level;
- No descendant of a hierarchical element is at the same level as that hierarchical element; and
- At this point, we don't care about the ordering of the levels.

From these assumptions and the information models, we can define FFD using the procedural constructs of EXPRESS. Specifically, a function (FFD) can be specified as in Figure 4. In the process of defining this one metric, several issues were identified. These issues are discussed in detail in [12].

Ties to Maintenance Feedback

In 1993, a Project Authorization Request (PAR) was submitted to the IEEE for new standards project related to specifying information and services for test and maintenance information feedback. The Test and Maintenance Information Management Standard (TMIMS) project was approved by the IEEE in early 1994. The focus of this project was to define exchange and service standards (similar to AI-ESTATE) which support the test and diagnostic maturation process. In 1998, due to a lack of progress, the TMIMS PAR was cancelled.

AI-ESTATE continues to require definition of exchange and service standards related to test and maintenance information. In 1998, shortly after the cancellation of the TMIMS PAR, the AI-ESTATE committee decided to include test and maintenance information in its scope. The approach will be consistent with AI-ESTATE (i.e., the definition of information models and EXPRESS-level services derived from traversing the models). Further, it is anticipated that the starting point for the new models will be the dynamic context model in IEEE 1232.2. By keeping track of the sequence of events during a diagnostic session, much of the historical information is identified and captured that can be used for later diagnostic maturation.

```
FUNCTION ffd(model:EDIM.edim; lvl:CEM.level) : REAL;
      LOCAL
            diag_count : INTEGER;
            diags : SET [0:?] OF EDIM.inference
            detect_set : SET [0:?] OF CEM.diagnosis := NULL;
      END_LOCAL;

      diag_count := SIZEOF(QUERY(tmp <* model.model_diagnosis |
      tmp.level_of_diagnosis = lvl);
      REPEAT I := LOINDEX(model.inference) TO HIINDEX(model.inference);
            diags := QUERY(tmp <* model.inference[I].conjuncts |
                        (TYPEOF(tmp) = 'EDIM.diagnostic_inference'));
            diags := diags + QUERY(tmp <* model.inference[i].disjuncts |
                        (TYPEOF(tmp) = 'EDIM.diagnostic_inference'));
            diags := QUERY(tmp <* diags |
                        tmp.pos_neg = negative OR
                        NOT(tmp.diagnostic_assertion = 'Good'));
            detect_set := detect_set + QUERY(tmp <* diags.for_diagnosis |
                        tmp.level_of_diagnosis = lvl);
      END_REPEAT;
      RETURN(SIZEOF(detect_set) / diag_count);

END_FUNCTION;
```

Figure 4. Sample Metric Definition in EXPRESS

Ties to Product Descriptions

Through the 1990s, the IEEE has been developing a family of standards under the umbrella of "A Broad Based Environment for Test" (ABBET) [13], [14]. An early architecture of ABBET, based on information modeling, presented ABBET as five layers: 1) product description, 2) test requirements/strategy, 3) test methods, 4) test resources, and 5) instrumentation. Since then, standards for the "lower layers" of ABBET (i.e., layers 3–5) have been defined; however, it has long been recognized that the major benefit from standardization will come from the "upper layers" (i.e., layers 1 and 2).

AI-ESTATE satisfies many of the requirements related to layer two of ABBET (however, AI-ESTATE has never been considered part of the ABBET family). Further, a recent proposal for a new information model-based standard, called the Test Requirements Model (TeRM), will address specific concerns of test requirements [15], [16]. Standards for the product description layer have always been problematic due to issues related to the revelation of intellectual property. With the combination of TeRM, AI-ESTATE, and TMIMS, it is anticipated that intellectual property can be hidden from information provided in standard form while still supporting the test engineer fully.

CONCLUSION

Reasoning system technology has progressed to the point where electronic and other complex systems are employing artificial intelligence as a primary component in meeting system test and verification requirements. This is giving rise to a proliferation of AI-based design, test, and diagnostic tools. Unfortunately, the lack of standard interfaces between these reasoning systems has increased the likelihood of significantly higher product life-cycle cost. Such costs arise from redundant engineering efforts during design and test phases, sizeable investment in special-purpose tools, and loss of system configuration control.

The AI-ESTATE standards promise to facilitate ease in production testing and long-term support of systems, as well as reducing overall product life-cycle cost. This will be accomplished by facilitating portability, knowledge reuse, and sharing of test and diagnostic information among embedded, automatic, and stand-alone test systems within the broader scope of product design, manufacture, and support.

AI-ESTATE was first conceived in 1988 as a standard for representing expert-system rule bases in the context of maintenance data collection. Since that time, AI-ESTATE has evolved to be embodied in three published standards related to the exchange of diagnostic information and the interaction of diagnostic reasoners within a test environment. The three standards have been recommended for inclusion on the US DoD ATS Executive Agent's list of standard satisfying requirements for ATS critical interfaces. In looking to the next generation, AI-ESTATE is expanding to address issues of testability, diagnosability, maintenance data collection, and test requirements specification.

ACKNOWLEDGMENTS

In many ways, it is unfortunate that a paper such as this includes only the names of two authors. The work reported here is the result of efforts of a committee of devoted volunteers who have supplied their expertise in system test and diagnosis to develop strong, sound standards supporting the diagnostics community. We would like to thank Les Orlidge, Randy Simpson, Tim Bearse, Tim Wilmering, Greg Bowman, Dave Kleinman, Lee Shombert, Sharon Goodall, Len Haynes, Jack Taylor, and Helmut Scheibenzuber for their efforts in developing the standards and promoting their use in industry.

REFERENCES

[1] IEEE Std 1232-1995. *IEEE Standard for Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE): Overview and Architecture*, Piscataway, NJ: IEEE Standards Press.

[2] IEEE Std 1232.2-1998. *IEEE Trial-Use Standard for Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE): Service Specification*, Piscataway, NJ: IEEE Standards Press.

[3] IEEE Std 1232.1-1997. *IEEE Trial-Use Standard for Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE): Data and Knowledge Specification*, Piscataway, NJ: IEEE Standards Press.

[4] ISO 10303-11:1994. *Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 11: Description Methods: The EXPRESS Language Reference Manual*, Geneva, Switzerland: International Organization for Standardization.

[5] ISO 10303-12:1997. *Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 12: Description Methods: The EXPRESS-I Language Reference Manual*, Geneva, Switzerland: International Organization for Standardization.

[6] Schenk, D. and Wilson, P. 1994. *Information Modeling: The EXPRESS Way*, New York: Oxford University Press.

[7] Sheppard, J. and Maguire, R. 1996. "Application Scenarios for AI-ESTATE Services," *AUTOTESTCON '96 Conference Record*, New York: IEEE Press.

[8] Sheppard, J. and Orlidge, L. 1997. Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE)—A New Standard for System Diagnostics," *Proceedings of the International Test Conference*, Los Alamitos, CA: IEEE Computer Society Press.

[9] Simpson, W. and Sheppard, J. 1994. *System Test and Diagnosis*, Boston, MA: Kluwer Academic Publishers.

[10] MIL STD 2165. 1985. *Testability Program for Electronic Systems and Equipment*, Washington, DC: Naval Electronic Systems Command (ELEX-8111).

[11] Perry, William. 1994. "Specifications and Standards—A New Way of Doing Business," US Department of Defense Policy Memorandum.

[12] Kaufman, M. and Sheppard, J. 1999. "IEEE P1522—A New Standard in Testability and Diagnosability Assessment, *AUTOTESTCON '99 Conference Record*, New York: IEEE Press.

[13] IEEE Std 1226-1993. *IEEE Trial-Use Standard for A Broad Based Environment for Test (ABBET): Overview and Architecture*, Piscataway, NJ: IEEE Standards Press.

[14] IEEE Std 1226.6-1996. *IEEE Guide to the Understanding of A Broad Based Environment for Test (ABBET)*, Piscataway, NJ: IEEE Standards Press.

[15] Shombert, L. 1998. *Test Requirements Model Language Reference Manual*, Draft 0.1, Technical Report CAE-1998-07-01, Vienna, VA: Intermetrics.

[16] Shombert, L. and Sheppard, J. 1998. "A Behavior Model for Next Generation Test Systems," *Journal of Electronic Testing: Theory and Applications*, Vol. 13, No. 3.